ProbLog4Fairness: A Neurosymbolic Approach to Modeling and Mitigating Bias

Rik Adriaensen[*1], **Lucas Van Praet**[*1], **Jessa Bekker**[1], **Robin Manhaeve**[1], **Pieter Delobelle**[1,2], **Maarten Buyl**[3]

[1]Department of Computer Science; Leuven.AI, KU Leuven, Belgium
[2]Aleph Alpha GmbH, Heidelberg, Germany
[3]IDLab, Ghent University, Belgium
{rik.adriaensen, lucas.vanpraet}@kuleuven.be

Abstract

Operationalizing definitions of fairness is difficult in practice, as multiple definitions can be incompatible while each being arguably desirable. Instead, it may be easier to directly describe algorithmic bias through ad-hoc assumptions specific to a particular real-world task, e.g., based on background information on systemic biases in its context. Such assumptions can, in turn, be used to mitigate this bias during training. Yet, a framework for incorporating such assumptions that is simultaneously principled, flexible, and interpretable is currently lacking. Our approach is to formalize bias assumptions as programs in ProbLog, a probabilistic logic programming language that allows for the description of probabilistic causal relationships through logic. Neurosymbolic extensions of ProbLog then allow for easy integration of these assumptions in a neural network's training process. We propose a set of templates to express different types of bias and show the versatility of our approach on synthetic tabular datasets with known biases. Using estimates of the bias distortions present, we also succeed in mitigating algorithmic bias in real-world tabular and image data. We conclude that *ProbLog4Fairness* outperforms baselines due to its ability to flexibly model the relevant bias assumptions, where other methods typically uphold a fixed bias type or notion of fairness.

Code — https://github.com/ML-KULeuven/PL4Fairness
Extended version — arxiv.org/abs/2511.09768

Introduction

One way to address fairness concerns is by including a fairness constraint in the learning process (Zafar et al. 2017; Cruz and Hardt 2024). However, there is no universally accepted definition of fairness, resulting in a large variety of fairness constraints (Verma and Rubin 2018), some of which are mutually incompatible (Kleinberg, Mullainathan, and Raghavan 2016; Chouldechova 2017; Defrance and De Bie 2023). The choice of fairness constraint can therefore be normatively contentious (Friedler, Scheidegger, and Venkatasubramanian 2021). Moreover, while enforcing or optimizing for the satisfaction of these constraints can be effective in improving the corresponding fairness metric, it lacks the nuance to holistically consider all forms of bias in a specific use case (Selbst et al. 2019; Buyl and De Bie 2024).

Alternatively, we can focus directly on addressing the factors that may cause algorithms to be unfair. We assume a significant contributor to such unfairness is the presence of *biasing mechanisms* in the data generation process, as models trained on biased data typically inherit these biases (Mehrabi et al. 2021). If we can explicitly model and adjust for these mechanisms during the learning process, the resulting model may 'naturally' produce fairer decisions, without ever needing to decide on a specific fairness constraint to optimize.

For example, consider a model that rates the creditworthiness of loan applicants. We may observe that the model scores applicants less favorably in correlation with a certain *sensitive variable* (e.g., ethnicity or gender), which could be considered indirect discrimination (Hacker 2018). This could arise, for instance, because the applicant's place of residence is used in the creditworthiness estimation. However, to what extent is such a correlation justifiable, e.g., because people in the disadvantaged areas are indeed less likely to repay their loans, and how much may this correlation reflect stereotyping, e.g., against an ethnicity dominant in that region? To address these questions, we consider the different mechanisms that may lead to such correlations. If our data consists of labels assigned by human annotators who themselves estimate creditworthiness, they may introduce *label bias* by undervaluing certain groups. We also consider *measurement bias* and *historical bias*, but our method is not limited to these three mechanisms.

These mechanisms and the data generation process are often considered probabilistic. Thus, it is natural to describe assumptions about them using probabilistic causal models, often represented as Bayesian networks (BNs). However, practitioners report that such models can be difficult to interpret and integrate into data pipelines (Ferrara et al. 2024). Hence, we use ProbLog to declaratively specify the relevant BN in a principled, flexible, and interpretable way using logical facts and rules. DeepProbLog (Manhaeve et al. 2021), an extension of ProbLog that allows the parameters of the BN to be predicted by neural networks, then facilitates the integration of these assumptions in the training process of a classifier. Concretely, this work contributes:

- A set of ProbLog templates for describing label, measurement, and historical biasing mechanisms, and an ap-

[*]These authors contributed equally.

proach for integrating these mechanisms into the training process of a classifier using DeepProbLog.

- Experiments demonstrating the ability of our method to flexibly model the correct bias assumption, thereby significantly improving both fairness and accuracy on unbiased labels, despite training on biased data.

Related Work

Research on fairness in machine learning often focuses on quantifying fairness with a metric and then either preprocessing the dataset (e.g., Feldman et al. 2015; Kamiran and Calders 2012), changing the model's optimization process (inprocessing) (e.g., Kamishima, Akaho, and Sakuma 2011; Delobelle et al. 2021), or postprocessing the predictions (e.g., Hardt, Price, and Srebro 2016) to improve the chosen metric.

Some works have used causal models to identify how a sensitive variable affects other features and the final decision (Calders and Verwer 2010; Kilbertus et al. 2017; Madras et al. 2018; Kusner et al. 2017, *inter alia*). For example, Kilbertus et al. (2017) model the generation process of biased data as a causal graph and provide a procedure to remove a specific kind of discrimination. Madras et al. (2018) model a causal graph containing the decision and its effect, with the sensitive variable as a confounding variable. Based on that graph, they infer the effects of other hidden confounders and learn a fairer decision policy.

Neurosymbolic techniques for fairness build on the idea of modeling and learning causal graphs to obtain fair classifiers. Varley and Belle (2021) introduce an algorithm that learns dependencies between sensitive variables and other variables using a sum-product network and then preprocesses the dataset to remove these dependencies. Choi, Dang, and den Broeck (2020) define a new probabilistic circuit structure that inherently satisfies a set of independence assumptions. Wagner and d'Avila Garcez (2021) develop a neurosymbolic active learning method based on Logic Tensor Networks and explainability methods, where a user queries a trained Logic Tensor Network or tests Shapley values to automatically add constraints to the model that are optimized during retraining.

Recently, Verreet, De Raedt, and Bekker (2024) have shown that a variety of labeling mechanisms, rules that describe the labeling process, from the Positive-Unlabeled (PU) learning setting can be expressed as templates in ProbLog. They show that by expressing their underlying assumptions as ProbLog rules, they generalize other methods.

Algorithmic Fairness

Notation Random variables are written as uppercase letters, with the corresponding lowercase letter denoting a specific assignment. Letters in boldface denote vectors. We use the $\tilde{V}$ notation to indicate a biased version of the variable V, where observations of the actual V are typically unavailable.

The binary variable A denotes the sensitive variable (i.e, the variable we want to avoid discrimination against), where $A = 1$ is the sensitive group. The vector $\mathbf{X}$ represents the feature vector, which includes the sensitive variable. Binary variable Y denotes the unbiased label, where $Y = 1$ is favorable. The predicted label is represented by $\hat{Y}$.

Defining fairness. Algorithmic fairness is a complex, multi-faceted, and often context-dependent problem with no general solutions (Selbst et al. 2019). Here, we consider the technical challenge of mitigating bias to achieve fair predictions. We introduce some key concepts and refer to Caton and Haas (2024) for further background.

A common approach to algorithmic fairness is to enforce fairness constraints. For example, *statistical parity*

$$P(\hat{Y} = 1 \mid A = 1) = P(\hat{Y} = 1 \mid A = 0) \tag{1}$$

which asserts that both groups have the same positive rate in expectation, or *equalized odds*, which checks whether the true and false positive rates are equal.

Statistical *dis*parity, the difference between the mean positive probability assigned to the sensitive and non-sensitive group, is a measure of fairness derived from this first constraint.

Fair classifiers. Ignoring fairness considerations, the goal in a binary classification setting is to learn a classifier $h^* : \mathrm{dom}(\mathbf{X}) \rightarrow [0, 1]$ predicting the conditional probability $P(\hat{Y} \mid \mathbf{X})$ that minimizes the prediction error $P(\hat{Y} \neq Y)$. For example, a model trained to minimize binary cross-entropy will approach h^*, given sufficient data and capacity.

However, we consider the scenario in which unbiased features $\mathbf{x}$ and/or labels y are unavailable during training. In practice, datasets are often biased due to, for example, more labeling errors or lower-quality data for certain groups, as well as systemic biases in the past. Therefore, we assume that we observe features $\tilde{\mathbf{x}}$ and/or labels $\tilde{y}$ drawn from a biased distribution. Training a classifier to minimize prediction error on this biased data would result in the classifier inheriting these unfair patterns or producing lower-quality predictions for a disadvantaged group.

Therefore, our objective is to learn a function h that minimizes the prediction error as if it were measured on the unbiased dataset, even though we only have biased features $\tilde{\mathbf{x}}$ and/or labels $\tilde{y}$ available during training. To achieve this, we assume that the *biasing mechanism*, the mapping between biased and unbiased variables, can be modeled as a BN.

At test time, it is always the goal to predict unbiased labels. However, we may or may not have access to the unbiased features. Therefore, we explicitly distinguish between two scenarios, where predictions are made on either (1) unbiased features at test time or (2) biased features at test time.

Deep Probabilistic Logic Programming

Our method uses *Deep Probabilistic Logic Programming*, an approach to neurosymbolic AI that combines logic programming, probability theory, and neural networks.

Logic Programming

Logic programming is a declarative way of describing a problem in terms of a logical theory, called a logic program, and a query. The task is to determine the truth value of the query within the program. We introduce some fundamental

concepts of logic programming here and refer to Flach and Sokol (2022) for further background.

A logic program consists of a set of rules. Rules are expressions of the form $\mathsf{h} :- \mathsf{b}_1, ..., \mathsf{b}_k$, where h is a logical atom and b_i are literals, i.e., atoms or their negation. The head h is true if all b_i's in the body are true. For example, $\mathsf{wet} :- \mathsf{exposed}, \mathsf{raining}$ is a rule expressing that a place is wet if it is exposed and it is raining. A rule with an empty body (e.g., $\mathsf{raining}$) is a fact and automatically true. In general, atoms are expressions of the form $\mathsf{q}(\mathsf{t}_1, ..., \mathsf{t}_k)$ potentially containing terms as arguments. These terms can be variables to allow for more general expressions. For example, $\mathsf{wet}(\mathsf{P}) :- \mathsf{exposed}(\mathsf{P}), \mathsf{raining}$ expresses that any person P who is exposed will get wet when it is raining. When proving a query, P will be substituted by a specific person.

ProbLog

ProbLog (De Raedt and Kimmig 2015) is an extension of the logic programming language Prolog with probabilistic facts $\mathsf{p} :: \mathsf{f}$, where f is a fact without variables and p is the probability of that fact being true. A probabilistic rule $\mathsf{p} :: \mathsf{h} :- \mathsf{b}$ represents that h has a probability p of being true *if* b is true, thus expressing a conditional probability.

Example 1 *As an example, consider this ProbLog program modeling a form of label bias.*

$\mathsf{poor\_neighborhood}(\mathsf{mary}).$

$\mathsf{can\_pay\_loan}(\mathsf{mary}).$

$\mathsf{can\_pay\_loan}(\mathsf{john}).$

$0.1 :: \mathsf{neg\_bias}(\mathsf{A}) :- \mathsf{poor\_neighborhood}(\mathsf{A}).$

$\mathsf{gets\_loan}(\mathsf{A}) :- \mathsf{can\_pay\_loan}(\mathsf{A}), \neg\mathsf{neg\_bias}(\mathsf{A}).$

An applicant A *is given a loan if they can pay it back and there is no negative bias, i.e., a bias causes* 10% *of all people in a poor neighborhood who can repay a loan to be rejected.*

In ProbLog, every probabilistic fact corresponds to an independent Bernoulli random variable that is true with probability p. This induces a probability distribution over all subsets of facts, each of which is called a possible world. The success probability of a query (e.g., $\mathsf{gets\_loan}(\mathsf{mary})$) is then defined as the expected probability that a possible world entails that query. In Example 1, the probability of $\mathsf{gets\_loan}(\mathsf{mary})$ is 0.9, while the probability of $\mathsf{gets\_loan}(\mathsf{john})$ is 1.0.

DeepProbLog

DeepProbLog (Manhaeve et al. 2021) extends ProbLog by allowing the labels of probabilistic facts to be predicted by neural networks, which enables reasoning on top of a neural network's predictions. Example 2 shows a neural network $\mathsf{h}(\mathbf{X})$ predicting the probability of a label $\mathsf{y}_\mathsf{h}(\mathbf{X})$, where $\mathbf{X}$ will be substituted by a specific feature vector $\mathbf{x}$. In Example 1, for instance, a classifier could predict the probability of $\mathsf{can\_pay\_loan}(\mathsf{mary})$.

Example 2 *A classifier as a probabilistic fact.*

$$\mathsf{h}(\mathbf{X}) :: \mathsf{y}_\mathsf{h}(\mathbf{X}).$$

In practice, the logic program is compiled into a circuit that computes the probability of a queried fact, e.g., $\mathsf{can\_pay\_loan}(\mathsf{mary})$, given the network's predictions. Importantly, the gradients of distant supervision on entailed facts, e.g., $\mathsf{gets\_loan}(\mathsf{mary})$, can be propagated back through this circuit, which allows the network to be trained with standard gradient-based approaches.

ProbLog Bias Modeling and Mitigation

To mitigate algorithmic bias, we now propose a two-step method: (1) model the biasing mechanism as a ProbLog program, and (2) integrate it into the classifier's training with DeepProbLog's distant supervision to mitigate the bias.

We define a logic program containing both a classifier for predicting the unbiased label $\mathsf{y}_\mathsf{h}(\mathbf{X})$ from the unbiased features $\mathbf{X}$ as in Example 2 and a biasing mechanism to represent the probabilistic transformation between unbiased features/labels and observed (potentially biased) features/labels. This allows the classifier to learn, from a biased dataset, how to make unbiased classifications, while the mechanism does the transformations between unobserved and observed variables. The classifier is trained using distant supervision: the program is supervised only through the observed biased labels and features, while DeepProbLog backpropagates gradients through the logic to update the network. This results in gradient updates according to all possible unbiased interpretations consistent with the observed biased data.

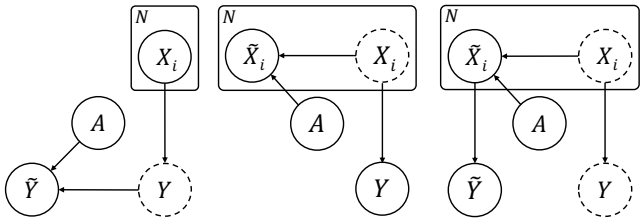
If the unbiased features are available at test time, we can drop the logic and use $\mathsf{h}(\mathbf{X})$ directly to make unbiased classifications from the unbiased features. However, if only biased features are available at test time, we keep the mechanism transforming biased to unbiased features.

DeepProbLog thus offers a ready-to-use framework for mitigating any biasing mechanism modeled in ProbLog, including settings with non-tabular data. This strategy, called *ProbLog4Fairness*, is more interpretable than causal fairness methods and allows for flexibly adding or revising assumptions where necessary rather than relying on fully specified graphs. In the remainder of this section, we discuss how to model different types of bias.

Modeling the Biasing Mechanism

Although any bias representable as a BN can be modeled in ProbLog (De Raedt and Kimmig 2015), we focus on three common biasing mechanisms depicted in Figure 1 to showcase our approach. We assume that the unbiased variables (X_i and Y) are independent of A, unless stated otherwise. Since A in that case does not help in predicting Y, we generally omit it from the input of a classifier.

Bias as probabilistic facts. A biasing mechanism describes the probabilistic transformation between the biased and unbiased distributions of a variable given the sensitive variable. For binary variables, this transformation is fully defined by four probabilities, essentially defining a conditional probability table. Therefore, we represent bias using four probabilistic facts with probabilities depending on (1) whether the bias affects the value negatively (i.e., changes 1 to 0) or positively (i.e., changes 0 to 1), and (2) whether the



(a) Label (b) Measurement (c) Historical

Figure 1: The Bayesian networks for label, measurement, and historical bias. Dashed nodes are unobserved.

person is part of the sensitive group. Example 3 shows these facts for label bias, where variable $\mathbf{X}$ will be substituted for a specific feature vector $\mathbf{x}$. The atom $\mathsf{a}(\mathbf{x})$ is then true when $A = 1$ in that feature vector.

If we assume a bias case never occurs, we can set that probability to 0 or remove the rule. In Example 1, we only used the first rule with $\mathsf{p}_1 = 0.1$ to represent the probability of negative discrimination against applicants from a poor neighborhood. Similarly, p_3 in the third rule would represent the probability of someone from a poor neighborhood receiving a loan they cannot repay.

The parameters p_i can be either set using domain knowledge or estimated from a small subset of the data with both biased and unbiased labels, as demonstrated in the experiments. When the parameters are learned jointly, however, the optimal classifier becomes unidentifiable because the model that minimizes the loss is no longer unique. This is a well-known problem in the PU learning literature (Gerych et al. 2022). Since these new solutions could contain biased classifiers, we consider this setting to be out of scope.

To model bias on a multiclass label or with a categorical sensitive variable, we could define a probabilistic fact for every possible transformation between values of the label and a rule for every value of the sensitive variable.

Example 3 *The probabilistic facts for label bias.*

$$\mathsf{p}_1 :: \mathsf{label\_neg\_bias}(\mathbf{X}) :- \mathsf{a}(\mathbf{X}).$$
$$\mathsf{p}_2 :: \mathsf{label\_neg\_bias}(\mathbf{X}) :- \neg\mathsf{a}(\mathbf{X}).$$
$$\mathsf{p}_3 :: \mathsf{label\_pos\_bias}(\mathbf{X}) :- \mathsf{a}(\mathbf{X}).$$
$$\mathsf{p}_4 :: \mathsf{label\_pos\_bias}(\mathbf{X}) :- \neg\mathsf{a}(\mathbf{X}).$$

Label Bias. In the case of label bias (Figure 1a), the unbiased features $\mathbf{X}$ are available, but the observed label $\tilde{Y}$ is a biased and noisy proxy of the unobserved fair label Y. In expectation, the labels $\tilde{Y}$ are less favorable for the sensitive group than the original labels Y. This could occur when loan applications are more often rejected because a loan officer directly discriminates against a sensitive variable, such as ethnicity. The biasing mechanism, i.e., the transformation from Y to $\tilde{Y}$, is now fully defined by the parameters $P(\tilde{Y} \mid Y = y, A = a)$ for all y and a, as they describe the full conditional probability table of *the biased labels given the unbiased labels*. Template 1 shows the label biasing mechanism in ProbLog, with the classifier and bias probabilities as described in Examples 2 and 3. The biased prediction $\tilde{\mathsf{y}}$ is true if y_h is false and the label is positively biased, or if y_h is true and the label is not negatively biased.

ProbLog Template 1 *Biased prediction under label bias.*

$$\tilde{\mathsf{y}}(\mathbf{X}) \ :- \ \neg\mathsf{y}_\mathsf{h}(\mathbf{X}), \mathsf{label\_pos\_bias}(\mathbf{X}).$$
$$\tilde{\mathsf{y}}(\mathbf{X}) \ :- \ \mathsf{y}_\mathsf{h}(\mathbf{X}), \neg\mathsf{label\_neg\_bias}(\mathbf{X}).$$

A concrete program can model multiple mechanisms that affect the label, or it can make stricter assumptions, for example, by assuming the biased labels are always correct for the non-sensitive group. This lowers the number of parameters needed to describe the full conditional probability table and simplifies the mechanism, as it is equivalent to removing lines 2 and 4 in Example 3. For instance, Example 4 shows a loan application case in which the only source of bias is a negative bias for the sensitive group, and there is noise on the label due to random mistakes with probability $\mathsf{p}_\mathsf{noise}$.

Example 4 *A specific model for label bias in a loan application.*

$$\mathsf{h}(\mathbf{X}) :: \mathsf{can\_pay\_loan}(\mathbf{X}).$$
$$0.21 :: \mathsf{neg\_bias}(\mathbf{X}) :- \mathsf{poor\_neighborhood}(\mathbf{X}).$$
$$\mathsf{gets\_loan}(\mathbf{X}) \ :- \ \mathsf{can\_pay\_loan}(\mathbf{X}), \neg\mathsf{neg\_bias}(\mathbf{X}).$$
$$\mathsf{p}_\mathsf{noise} :: \mathsf{noise}(\mathbf{X}).$$
$$\mathsf{observed\_gets\_loan}(\mathbf{X}) \ :- \ \mathsf{gets\_loan}(\mathbf{X}), \neg\mathsf{noise}(\mathbf{X}).$$
$$\mathsf{observed\_gets\_loan}(\mathbf{X}) \ :- \ \neg\mathsf{gets\_loan}(\mathbf{X}), \mathsf{noise}(\mathbf{X}).$$

Measurement Bias. When there is measurement bias (Figure 1b), the observed features are noisy and biased proxies of the unbiased features, but the label only depends on the unbiased features. Unfairness arises when these proxy features are often less favorable or noisier for the sensitive group. For example, measurement bias occurs when 'days worked in the past three years' is used as a proxy for job stability, which is negatively biased against women who took maternity leave.

The parameters describing the bias are now specified by the conditional probability table of some biased feature $\tilde{X}_i$ given the unbiased feature X_i, i.e., $P(X_i \mid \tilde{X}_i = \tilde{x}_i, A = a)$ for all $\tilde{x}_i$ and a. The probabilistic facts follow the same structure as in Example 3 but describe the transformation *from a biased to an unbiased feature*, opposite to before. The mechanism for predicting the unbiased label under measurement bias is given in Template 2 for a feature vector with biased feature b.

ProbLog Template 2 *Fair prediction under measurement bias.*

$$\mathsf{b\_biased}(\tilde{\mathbf{X}}) \ :- \ \neg\mathsf{b}(\tilde{\mathbf{X}}), \mathsf{b\_neg\_bias}(\tilde{\mathbf{X}}).$$
$$\mathsf{b\_biased}(\tilde{\mathbf{X}}) \ :- \ \mathsf{b}(\tilde{\mathbf{X}}), \mathsf{b\_pos\_bias}(\tilde{\mathbf{X}}).$$
$$\mathsf{debias}(\tilde{\mathbf{X}}, \mathbf{X}) \ :- \ \mathsf{b\_biased}(\tilde{\mathbf{X}}), \mathsf{debias\_b}(\tilde{\mathbf{X}}, \mathbf{X}).$$
$$\mathsf{debias}(\tilde{\mathbf{X}}, \tilde{\mathbf{X}}) \ :- \ \neg\mathsf{b\_biased}(\tilde{\mathbf{X}}).$$
$$\mathsf{y}(\tilde{\mathbf{X}}) \ :- \ \mathsf{debias}(\tilde{\mathbf{X}}, \mathbf{X}), \mathsf{y}_\mathsf{h}(\mathbf{X}).$$

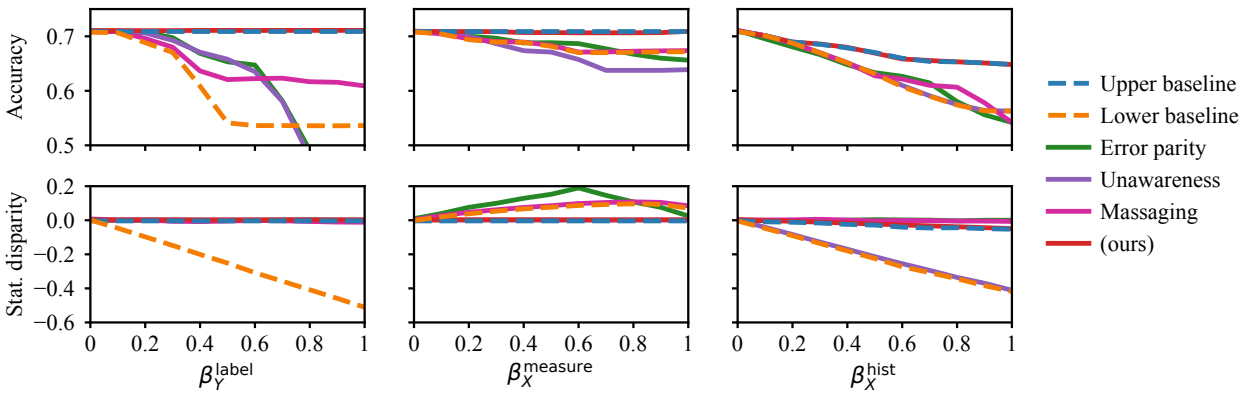


Figure 2: ProbLog4Fairness successfully models different types of bias, approaching the upper baseline. We measure accuracy and statistical disparity for an increasing probability of label (left), measurement (middle), and historical (right) bias during training while evaluating on unbiased data at test time.

The label prediction is the combination of the predictions on the different possible unbiased feature vectors, weighted by how probable they are given the biased feature vector. Predicate $\mathsf{b}(\tilde{\mathbf{X}})$ is true when the biased feature in the feature vector $\tilde{\mathbf{X}}$ is one and $\mathsf{debias\_b}(\tilde{\mathbf{X}}, \mathbf{X})$ substitutes variable $\mathbf{X}$ with a vector identical to $\tilde{\mathbf{X}}$ but with the value of b flipped. This approach to modeling bias on b can be easily extended to model multiple biased features.

Historical Bias. Historical bias (Figure 1c) occurs when the features in the dataset are biased and, in contrast to measurement bias, the labels are based on the biased features. Therefore, the bias is present in both the features and the label. For example, discrimination can lead to limited job opportunities, resulting in less favorable features (e.g., lower income or unemployment), which actually cause the sensitive group to default on loans more often. However, in a normatively desirable world without discrimination, loan repayment would be more consistent as these features would be more similarly distributed for the sensitive group. The transformation from the historically biased to the unbiased distribution can be modeled in the same way as measurement bias in Template 2. As the labels would also change in this desirable world, different from measurement bias, we can combine this with a transformation from unbiased to biased labels modeled as label bias in Template 1.

However, it is sometimes possible under historical bias to assume that the function from biased features to biased labels is the same as the one from unbiased features to unbiased labels. As a result, a classifier trained on the biased features and labels will make fair predictions if the features are unbiased at test time. Under this assumption, the bias is *only* a result of the unfavorable feature distribution for the sensitive group. Therefore, if we have biased features at test time, we can simply learn the fair classifier on the biased data and use the measurement biasing mechanism during test time to predict as if in a normatively desirable world. This can help counteract the historical disadvantages, for example, by providing people who were historically discriminated against with more opportunities despite having a lower income.

Experiments

We now verify our bias modeling and mitigation approach *ProbLog4Fairness* experimentally against synthetic data and two real-world datasets: one tabular and one image.

Baselines. All baselines use a neural network with the same architecture as a classifier. The *Lower baseline* is directly trained on the observed features and labels. The *Upper baseline* is directly trained on the unbiased features and labels. *Unawareness* is trained on the observed features and labels, but with omission of the sensitive variable (Dwork et al. 2012). *Massaging* preprocesses the biased dataset by demoting positive labels from the non-sensitive group and promoting negative labels from the sensitive group based on the ranking of a model fitted to the data. Afterwards, a classifier is trained on this massaged dataset (Kamiran and Calders 2009). Finally, *Error parity* is a postprocessing technique that adapts the lower baseline's predictions to satisfy a fairness constraint (Cruz and Hardt 2024). We use statistical parity as fairness constraint for this baseline.

Methodology. To measure predictive performance, we report accuracy or the F1 score when labels are imbalanced. We use statistical disparity to evaluate fairness. We tested using 5-fold cross-validation and repeated each experiment on at least five different seeds, reporting the mean.

Synthetic Data Experiments

First, we experiment on synthetic datasets where we have explicit control over the type and probability of bias. The data generation process we use is based on (Baumann et al. 2023), but restricted to binary and categorical features. The probability of label, measurement, and historical bias occurring is controlled by $\beta_{\mathbf{Y}}^{label}$, $\beta_{\mathbf{X}}^{measure}$, and $\beta_{\mathbf{X}}^{hist}$, respectively. If there is measurement or historical bias, the probability is assumed to be the same for each feature. For RQ1 and RQ2, we assume the relevant bias probability β is known and we set the parameters in our program accordingly. In RQ3, we investigate what to do when β is unknown.

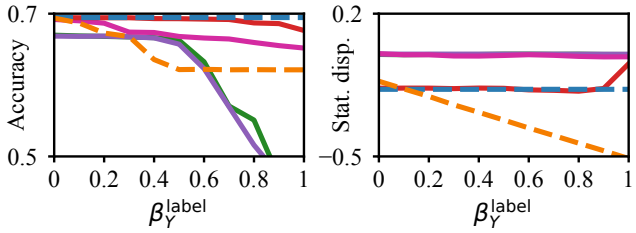


Figure 3: Our method is able to remove only the problematic bias when $A \not\perp Y$, approaching the upper baseline. We measure accuracy and statistical disparity on unbiased data while training with an increasing probability of label bias under $A \not\perp Y$. Color coding is consistent with the legend in Figure 2.

RQ1. How does ProbLog4Fairness compare to the baselines under different types of bias in terms of predictive performance and fairness? Figure 2 shows the accuracy and statistical disparity of our approach against the baselines under label bias (using Template 1 during training), measurement bias (using Template 2 during training) and historical bias (using Template 2 at test time) for an increasing bias probability. The models are tested on unbiased data. Given the correct parameters for the program, our approach achieves an accuracy and statistical disparity comparable to the upper baseline despite training on the biased labels. The mitigating baselines are generally effective in achieving fairness, but their predictive performance on the unbiased data is limited. Due to the flexibility of our programs, we are able to deal with a wide range of biases effectively.

RQ2. Can our approach mitigate bias when $A \not\perp Y$? ProbLog4Fairness is able to effectively mitigate only the problematic bias, as can be seen in Figure 3. By setting the parameters of our program based on the correct β, which does not capture the nonproblematic part of the correlation between A and Y, we achieve the statistical disparity of the upper baseline, which is no longer expected to be close to zero. In contrast, the lower baseline predicts a disproportionately smaller mean positive probability for the sensitive group as the probability of the bias increases. The other baselines are not able to make the distinction between problematic and nonproblematic bias and incorrectly impose a statistical disparity of zero.

RQ3. What is the effect of choosing the wrong parameters in the program? Figure 4 shows the statistical disparity and accuracy for our method, evaluated on unbiased labels, for a fixed label bias probability of 0.3 and $A \not\perp Y$, as we vary the bias probability $\hat{\beta}$ used to calculate the parameters of the program. The trained classifier achieves the highest accuracy and a statistical disparity closest to the upper baseline when the parameters in the program are chosen to match the actual bias probability. Importantly, around this optimum, the sensitivity for a parameter estimation error is small. Therefore, our method will be effective if we estimate these parameters from a limited subset of the data for which the unbiased features/labels are also available.

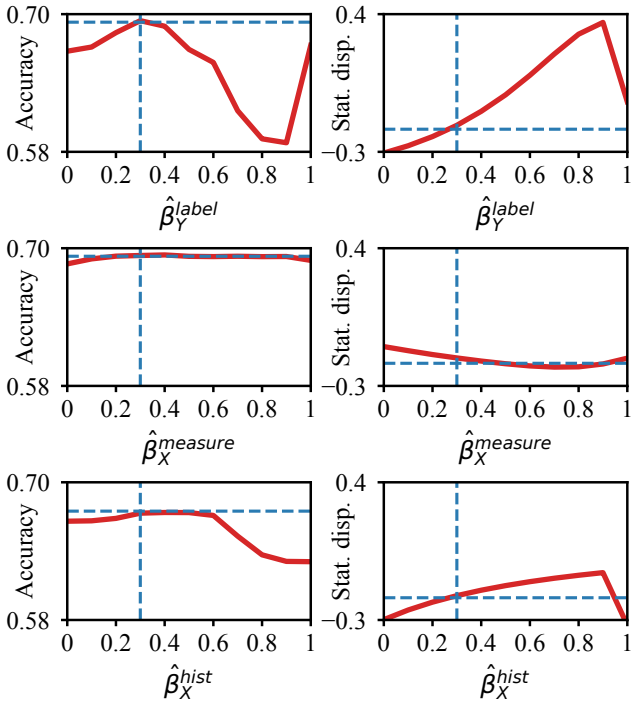


Figure 4: Our method achieves the highest accuracy and the expected statistical disparity when the correct bias probability is used to estimate the parameters in the program. We train on fixed bias probabilities of 0.3, with $A \not\perp Y$ and evaluate on unbiased data, but vary the bias probability $\hat{\beta}$ used to set the program's parameters. The dashed lines indicate the upper baseline.

Student Dataset Experiment

The *Student Alcohol Consumption* dataset introduced by Cortez and Silva (2008) contains tabular features about 856 students, such as their weekly alcohol consumption, gender, and how long they studied for an exam. The labels indicate whether they passed the exam and can be considered the unbiased labels (keeping any historical biases in exam success rates out of scope). Based on this data Lenders and Calders (2023) asked annotators to predict whether these students would pass based on their features. By priming the annotators, this produced biased labels against male students. This dataset is useful as the biased and unbiased labels are known, yet the bias is *real-world* in the sense that it results from the actual biased decisions of the human annotators.

RQ4. Can ProbLog4Fairness effectively mitigate real-world bias? We model the bias as label bias, with $A \perp Y$, and assume the bias probability is known. Figure 5 shows that our method approaches the statistical disparity present in the unbiased data and outperforms the baselines in terms of unbiased label F1 score. When evaluating on biased data, the baselines incorrectly appear to perform better, emphasizing the importance of evaluating on unbiased labels. As expected, we do not perform as close to the upper baseline for real-world data, likely because other biases are present

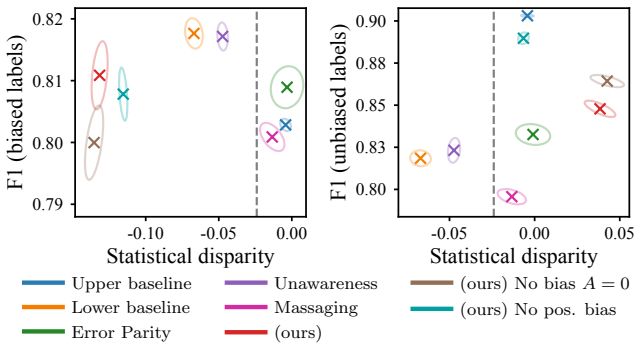


Figure 5: Our approach achieves a higher F1 score on the *unbiased* labels than the mitigating baselines and approaches the expected statistical disparity. Sensible simplifying assumptions seem to benefit performance. The gray vertical line indicates the statistical disparity in the unbiased labels. The ellipses show a 95% confidence region based on the standard error.

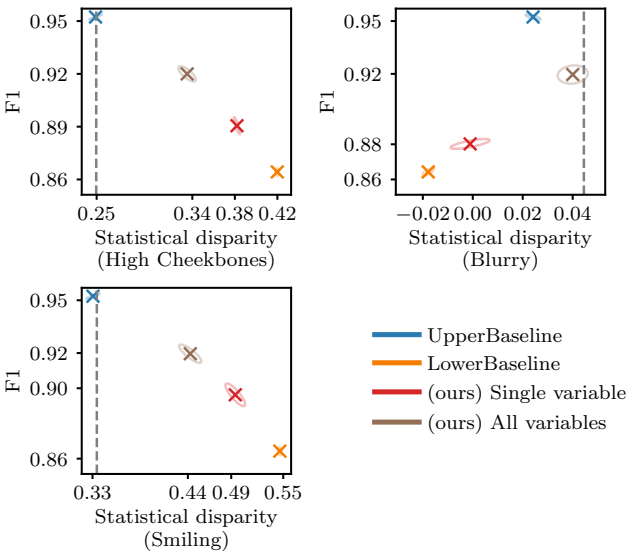


Figure 6: Our method successfully mitigates label bias for facial feature detection. Combining biasing mechanisms over multiple attributes benefits the overall F1 score on unbiased Mouth Slightly Open labels. The gray vertical line indicates the actual statistical disparity present in the unbiased data. The ellipses show a 95% confidence region based on the standard error.

that are not being accounted for. Therefore, we expect a more elaborate ProbLog program to achieve an even better F1 score.

RQ5. What is the effect of simplifying assumptions in the program on the predictive performance and fairness of our method? In Figure 5, we also show how two ProbLog programs with simplifying assumptions, 'no positive bias' and 'no bias on $A = 0$', perform. The simplified programs achieve a better F1 score and 'no positive bias' achieves a better statistical disparity. This could be due to a large estimation error on the positive bias parameters, as there are only 67 girls and 82 boys who did not pass and thus have an unbiased negative label. Another hypothesis favoring simple mitigation strategies is that under an expressive prior, a classifier could have difficulty learning what remains.

CELEB-A Experiment

The CELEB-A dataset (Liu et al. 2015) contains faces of celebrities with 40 binary attributes for each image, such as Smiling, Mouth Slightly Open, Blurry, and High Cheekbones. The aim is to predict these features from the image. Wu et al. (2023) show that these labels are often subjective and inconsistent between annotators, resulting in biased labels. They also provide a cleaned version of the Mouth Slightly Open attribute. We fine-tuned a ResNet-50 pretrained on ImageNet (Deng et al. 2009) on the cleaned labels as an upper baseline and achieved a 10.18% improvement in F1 score on the cleaned labels compared to fine-tuning on the original labels. Although fairness is less of a concern for this particular attribute, bias in facial recognition often does lead to unfair decisions when the label quality for a particular group is lower.

RQ6. Can ProbLog4Fairness effectively mitigate label bias on high-dimensional features, such as images? As shown in Figure 6, our method achieves part of the increase in F1 score of the upper baseline under the assumption of label bias on Mouth Slightly Open compared to three 'sensitive' variables, Smiling, Blurry, and High Cheekbones, that turn out to be correlated with the labeling errors. We also effectively progress the statistical disparity towards the level present in the unbiased data with respect to these three attributes. Notice, however, that correcting for all three variables at the same time does much better in terms of both F1 score and statistical disparity, compared to taking a single variable into account. This shows that eliminating the bias with respect to a single feature does not immediately imply fairness because other features correlated with the sensitive variable are not accounted for. This underlines the importance of being able to flexibly model the large number and variety of biases that can be present in a real-world context.

Conclusion

In conclusion, ProbLog4Fairness successfully models and mitigates various types of bias across synthetic and real-world datasets. Due to our ability to flexibly model the relevant bias assumptions, we outperform baselines that uphold a fixed assumption of bias or notion of fairness. Additionally, we show that the parameters in our program, if not available from background information, can be set using a small unbiased subset of the data.

Acknowledgements

This work was partially supported by the Interuniversity Special Research Fund iBOF/21/075 (KU Leuven - Universiteit Gent). This research also received funding from the Flemish Government (AI Research Program) and from

the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Advanced Grant DeepLog No. 101142702, Advanced Grant VIGILIA, No. 101142229). We are grateful to Luc De Raedt and Tijl De Bie for providing helpful feedback.

References

Baumann, J.; Castelnovo, A.; Crupi, R.; Inverardi, N.; and Regoli, D. 2023. Bias on Demand: A Modelling Framework That Generates Synthetic Data With Bias. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '23, 1002–1013. New York, NY, USA: Association for Computing Machinery. ISBN 9798400701924.

Buyl, M.; and De Bie, T. 2024. Inherent limitations of AI fairness. *Communications of the ACM*, 67(2): 48–55.

Calders, T.; and Verwer, S. 2010. Three naive bayes approaches for discrimination-free classification. *Data mining and knowledge discovery*, 21(2): 277–292.

Caton, S.; and Haas, C. 2024. Fairness in Machine Learning: A Survey. *ACM Comput. Surv.*, 56(7).

Choi, Y.; Dang, M.; and den Broeck, G. V. 2020. Group Fairness by Probabilistic Modeling with Latent Fair Decisions. In *AAAI Conference on Artificial Intelligence*.

Chouldechova, A. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2): 153–163.

Cortez, P.; and Silva, A. 2008. Using data mining to predict secondary school student performance. *EUROSIS*.

Cruz, A.; and Hardt, M. 2024. Unprocessing Seven Years of Algorithmic Fairness. In *The Twelfth International Conference on Learning Representations*.

De Raedt, L.; and Kimmig, A. 2015. Probabilistic (logic) programming concepts. *Machine Learning*, 100(1): 5–47.

Defrance, M.; and De Bie, T. 2023. Maximal fairness. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, 851–880.

Delobelle, P.; Temple, P.; Perrouin, G.; Frénay, B.; Heymans, P.; and Berendt, B. 2021. Ethical Adversaries: Towards Mitigating Unfairness with Adversarial Machine Learning. *SIGKDD Explor. Newsl.*, 23(1): 32–41.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255.

Dwork, C.; Hardt, M.; Pitassi, T.; Reingold, O.; and Zemel, R. 2012. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, 214–226.

Feldman, M.; Friedler, S. A.; Moeller, J.; Scheidegger, C.; and Venkatasubramanian, S. 2015. Certifying and removing disparate impact. In *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 259–268.

Ferrara, C.; Sellitto, G.; Ferrucci, F.; Palomba, F.; and De Lucia, A. 2024. Fairness-aware machine learning engineering: how far are we? *Empirical software engineering*, 29(1): 9.

Flach, P.; and Sokol, K. 2022. *Simply Logical – Intelligent Reasoning by Example (Fully Interactive Online Edition)*.

Friedler, S. A.; Scheidegger, C.; and Venkatasubramanian, S. 2021. The (im) possibility of fairness: Different value systems require different mechanisms for fair decision making. *Communications of the ACM*, 64(4): 136–143.

Gerych, W.; Hartvigsen, T.; Buquicchio, L.; Agu, E.; and Rundensteiner, E. 2022. Recovering the Propensity Score from Biased Positive Unlabeled Data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(6): 6694–6702.

Hacker, P. 2018. Teaching fairness to artificial intelligence: existing and novel strategies against algorithmic discrimination under EU law. *Common market law review*, 55(4).

Hardt, M.; Price, E.; and Srebro, N. 2016. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29.

Kamiran, F.; and Calders, T. 2009. Classifying without discriminating. In *Proceedings 2nd IEEE International Conference on Computer, Control and Communication (IC4 2009, Karachi, Pakistan, February 17-18, 2009)*, 1–6. United States: Institute of Electrical and Electronics Engineers. ISBN 978-1-4244-3313-1.

Kamiran, F.; and Calders, T. 2012. Data preprocessing techniques for classification without discrimination. *Knowledge and information systems*, 33(1): 1–33.

Kamishima, T.; Akaho, S.; and Sakuma, J. 2011. Fairness-aware learning through regularization approach. In *2011 IEEE 11th international conference on data mining workshops*, 643–650. IEEE.

Kilbertus, N.; Rojas Carulla, M.; Parascandolo, G.; Hardt, M.; Janzing, D.; and Schölkopf, B. 2017. Avoiding Discrimination through Causal Reasoning. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Kleinberg, J.; Mullainathan, S.; and Raghavan, M. 2016. Inherent trade-offs in the fair determination of risk scores. *arXiv preprint arXiv:1609.05807*.

Kusner, M. J.; Loftus, J.; Russell, C.; and Silva, R. 2017. Counterfactual fairness. *Advances in neural information processing systems*, 30.

Lenders, D.; and Calders, T. 2023. Real-life Performance of Fairness Interventions - Introducing A New Benchmarking Dataset for Fair ML. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, SAC '23, 350–357. New York, NY, USA: Association for Computing Machinery. ISBN 9781450395175.

Liu, Z.; Luo, P.; Wang, X.; and Tang, X. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Madras, D.; Creager, E.; Pitassi, T.; and Zemel, R. S. 2018. Fairness Through Causal Awareness: Learning Latent-Variable Models for Biased Data. *CoRR*, abs/1809.02519.

Manhaeve, R.; Dumančić, S.; Kimmig, A.; Demeester, T.; and De Raedt, L. 2021. Neural probabilistic logic programming in DeepProbLog. *Artificial Intelligence*, 298: 103504.

Mehrabi, N.; Morstatter, F.; Saxena, N.; Lerman, K.; and Galstyan, A. 2021. A survey on bias and fairness in machine learning. *ACM computing surveys (CSUR)*, 54(6): 1–35.

Selbst, A. D.; Boyd, D.; Friedler, S. A.; Venkatasubramanian, S.; and Vertesi, J. 2019. Fairness and abstraction in sociotechnical systems. In *Proceedings of the conference on fairness, accountability, and transparency*, 59–68.

Varley, M.; and Belle, V. 2021. Fairness in machine learning with tractable models. *Knowledge-Based Systems*, 215: 106715.

Verma, S.; and Rubin, J. 2018. Fairness definitions explained. In *Proceedings of the international workshop on software fairness*, 1–7.

Verreet, V.; De Raedt, L.; and Bekker, J. 2024. Modeling PU learning using probabilistic logic programming. *Machine Learning*, 113(3): 1351–1372.

Wagner, B.; and d'Avila Garcez, A. S. 2021. Neural-symbolic integration for fairness in AI. In *AAAI 2021 Spring Symposium on Combining Machine Learning and Knowledge Engineering (AAAI-MAKE 2021)*, volume 2846. ©2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

Wu, H.; Bezold, G.; Günther, M.; Boult, T.; King, M. C.; and Bowyer, K. W. 2023. Consistency and accuracy of celeba attribute values. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3258–3266.

Zafar, M. B.; Valera, I.; Rogriguez, M. G.; and Gummadi, K. P. 2017. Fairness Constraints: Mechanisms for Fair Classification. In Singh, A.; and Zhu, J., eds., *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, 962–970. PMLR.